

SIMLR: a tool for large-scale single-cell analysis by multi-kernel learning

Bo Wang* Daniele Ramazzotti* Luca De Sano*
Junjie Zhu Emma Pierson Serafim Batzoglou

Abstract

Motivation: We here present SIMLR (Single-cell Interpretation via Multi-kernel LeaRning), an open-source tool that implements a novel framework to learn a cell-to-cell similarity measure from single-cell RNA-seq data. SIMLR can be effectively used to perform tasks such as dimension reduction, clustering, and visualization of heterogeneous populations of cells. SIMLR was benchmarked against state-of-the-art methods for these three tasks on several public datasets, showing it to be scalable and capable of greatly improving clustering performance, as well as providing valuable insights by making the data more interpretable via better a visualization.

Availability and Implementation: SIMLR is available on GitHub in both R and MATLAB implementations. Furthermore, it is also available as an R package on bioconductor.org.

Contact: bowang87@stanford.edu or daniele.ramazzotti@stanford.edu

Supplementary Information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

The recent development of high resolution single-cell RNA-seq (scRNA-seq) technologies increases the availability of high throughput gene expression measurements of individual cells. This allows us to dissect previously unknown heterogeneity and functional diversity among cell populations [1]. In this line of work recent efforts (see [2, 3, 4]) have demonstrated that *de novo* cell type discovery of functionally distinct cell sub-populations is possible via unbiased analysis of all transcriptomic information provided by scRNA-seq data. However, such analysis heavily relies on the accurate assessment of pairwise cell-to-cell similarities, which poses unique challenges such as outlier cell populations, transcript amplification noise, and dropout events (*i.e.*, zero expression measurements due to sampling or stochastic transcriptional activities) [5].

Recently, new single-cell platforms such as DropSeq [6] and GemCode single-cell technology [7] have enabled a dramatic increase in throughput

*Equal contributors.

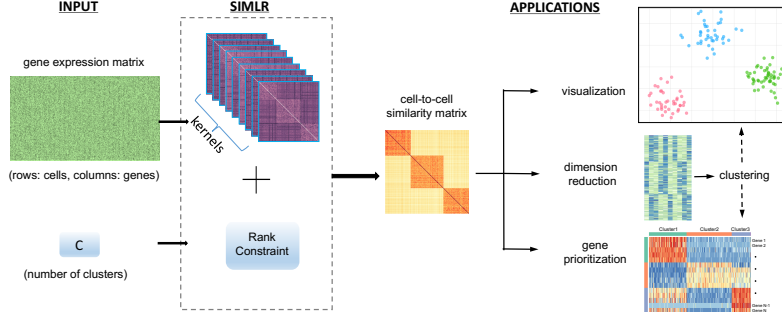


Figure 1: SIMLR pipeline [9]. We start with an input matrix with gene expression observations for a set of genes. SIMLR is then capable of learning a set of cell-to-cell similarities by estimating multiple kernels, with the assumptions of the presence of C separable populations within the data. To this extent, SIMLR constraints the similarity matrix to have an approximate block-diagonal structure with C blocks where the cells of the same populations to be more similar. The learned similarities can be used for multiple tasks; they can be used to visualize the cells, reduce the dimension of the data, cluster the cells into subgroups and prioritize the most variable genes that explain the differences across the populations.

to hundreds of thousands of cells. While such technological advances may add additional power for *de novo* discovery of cell populations, they also increase computational burdens for traditional unsupervised learning methods.

To address all of the aforementioned challenges, SIMLR was proposed in [8] as a novel framework that learns an appropriate cell-to-cell similarity metric from the input single-cell data. The learned similarities enable effective dimension reduction, clustering, and visualization of cells. SIMLR provides a more scalable analytical framework, which works on hundreds of thousands of cells without any loss of accuracy in dissecting cell heterogeneity.

2 The SIMLR framework

SIMLR is available in both R and MATLAB implementations. The framework is capable of learning cell-to-cell similarities among gene expression data of individual cells, which have been shown to capture different representations of the data. The approach combines multiple Gaussian kernels in an optimization framework, which can be efficiently solved by a simple iterative procedure. Moreover, SIMLR addresses the challenge of high levels of noise and dropout events by employing a rank constraint and graph diffusion in the learned cell-to-cell similarity [9]. See Figure 1 for an overview of the framework.

The framework provides both a standard implementation and a large-scale extension of SIMLR together with two examples to test the methods on the datasets by [10] for the standard SIMLR and [11] for the large-scale extension (see Supplementary Material for details). SIMLR can accurately analyze both datasets within minutes on a single core laptop.

One of the advantages of SIMLR is that the learned similarities can be efficiently adapted into multiple downstream applications. Some applications include prioritizing genes by ranking their concordance with the similarity and creating low-dimensional representations of cells by transforming the input into a stochastic neighbor embedding framework. We refer to the Supplementary Material for detailed use cases of the tool and to [8] for a detailed description of the method and for several applications on genomic data from public datasets.

3 Discussion

SIMLR infers the cell-to-cell similarities that are used to perform dimension reduction, clustering, and visualization. While the multiple-kernel learning framework has obvious advantages on heterogeneous single-cell datasets, where several clusters coexist, we also believe that this approach, together with its visualization framework, may also be valuable for data that does not contain clear clusters, such as cell populations that contain cells spanning a continuum or a developmental pathway.

References

- [1] Ehud Shapiro, Tamir Biezuner, and Sten Linnarsson. Single-cell sequencing-based technologies will revolutionize whole-organism science. *Nature Reviews Genetics*, 14(9):618–630, 2013.
- [2] Alex A Pollen, Tomasz J Nowakowski, Joe Shuga, Xiaohui Wang, Anne A Leyrat, Jan H Lui, Nianzhen Li, Lukasz Szpankowski, Brian Fowler, Peilin Chen, et al. Low-coverage single-cell mrna sequencing reveals cellular heterogeneity and activated signaling pathways in developing cerebral cortex. *Nature biotechnology*, 32(10):1053–1058, 2014.
- [3] Dmitry Usoskin, Alessandro Furlan, Saiful Islam, Hind Abdo, Peter Lönnerberg, Daohua Lou, Jens Hjerling-Leffler, Jesper Haeggström, Olga Kharchenko, Peter V Kharchenko, et al. Unbiased classification of sensory neuron types by large-scale single-cell rna sequencing. *Nature neuroscience*, 18(1):145–153, 2015.
- [4] Aleksandra A Kolodziejczyk, Jong Kyoung Kim, Jason CH Tsang, Tomislav Ilicic, Johan Henriksson, Kedar N Natarajan, Alex C Tuck, Xuefei Gao, Marc Bühler, Pentao Liu, et al. Single cell rna-sequencing of pluripotent states unlocks modular transcriptional variation. *Cell Stem Cell*, 17(4):471–485, 2015.
- [5] Emma Pierson and Christopher Yau. Zifa: Dimensionality reduction for zero-inflated single-cell gene expression analysis. *Genome biology*, 16(1):241, 2015.

- [6] Evan Z Macosko, Anindita Basu, Rahul Satija, James Nemesh, Karthik Shekhar, Melissa Goldman, Itay Tirosh, Allison R Bialas, Nolan Kamitaki, Emily M Mardersteck, et al. Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. *Cell*, 161(5):1202–1214, 2015.
- [7] Grace XY Zheng, Jessica M Terry, Phillip Belgrader, Paul Ryvkin, Zachary W Bent, Ryan Wilson, Solongo B Ziraldo, Tobias D Wheeler, Geoff P McDermott, Junjie Zhu, et al. Massively parallel digital transcriptional profiling of single cells. *Nature Communications*, 2017.
- [8] Bo Wang, Junjie Zhu, Emma Pierson, Daniele Ramazzotti, and Serafim Batzoglou. Visualization and analysis of single-cell rna-seq data by kernel- based similarity learning. *Nature Methods*, 2017.
- [9] Bo Wang, Aziz M Mezlini, Feyyaz Demir, Marc Fiume, Zhuowen Tu, Michael Brudno, Benjamin Haibe-Kains, and Anna Goldenberg. Similarity network fusion for aggregating data types on a genomic scale. *Nature methods*, 11(3):333–337, 2014.
- [10] Florian Buettner, Kedar N Natarajan, F Paolo Casale, Valentina Proserpio, Antonio Scialdone, Fabian J Theis, Sarah A Teichmann, John C Marioni, and Oliver Stegle. Computational analysis of cell-to-cell heterogeneity in single-cell rna-sequencing data reveals hidden subpopulations of cells. *Nature biotechnology*, 33(2):155–160, 2015.
- [11] Amit Zeisel, Ana B Muñoz-Manchado, Simone Codeluppi, Peter Lönnerberg, Gioele La Manno, Anna Juréus, Sueli Marques, Hermany Munguba, Liqun He, Christer Betsholtz, et al. Cell types in the mouse cortex and hippocampus revealed by single-cell rna-seq. *Science*, 347(6226):1138–1142, 2015.

SIMLR: a tool for large-scale single-cell analysis by multi-kernel learning

Supplementary Information

Bo Wang* Daniele Ramazzotti* Luca De Sano*
Junjie Zhu Emma Pierson Serafim Batzoglou

SIMLR is available in R and Matlab. We will now present details about each implementation.

1 R

1.1 Installation

The *SIMLR R package* for Single-cell Interpretation via Multi-kernel LeaRning is available on *Bioconductor* at <https://bioconductor.org/packages/release/bioc/html/SIMLR.html> and can be installed as follows.

```
## try http:// if https:// URLs are not supported
source("https://bioconductor.org/biocLite.R")
biocLite("SIMLR")
```

The package is also available on *Github* at <https://github.com/BatzoglouLabSU/SIMLR>. It is possible to install both the master (stable) and development versions of the R package by using the R library *devtools*.

```
library(devtools)
install_github("BatzoglouLabSU/SIMLR", ref = 'master')
library(SIMLR)

library(devtools)
install_github("BatzoglouLabSU/SIMLR", ref = 'development')
library(SIMLR)
```

1.2 Examples

We now show two use cases for SIMLR in order to highlight the main features of our tool. We first load the data provided as an example in the package. The

*Equal contributors.

dataset *BuettnerFlorian* [1] is used to illustrate how to use standard SIMLR, while a reduced version of the dataset *ZeiselAmit* [2] is used to illustrate how to use large-scale SIMLR.

```
library(SIMLR)
data(BuettnerFlorian)
data(ZeiselAmit)
```

The external R package *igraph* [3] is required for the computation of the normalized mutual information to assess the results of the clustering.

```
library(igraph)
```

We run SIMLR on the BuettnerFlorian input dataset. For this dataset we have a ground truth of 3 cell populations (clusters).

```
set.seed(11111)
example = SIMLR(X = BuettnerFlorian$in_X,
               c = BuettnerFlorian$n_clust)

## Computing the multiple Kernels.
## Performing network diffusion.
## Iteration: 1
## Iteration: 2
...
## Iteration: 10
## Iteration: 11
## Performing t-SNE.
## Epoch: Iteration # 100 error is: 0.1140084
## Epoch: Iteration # 200 error is: 0.06181848
...
## Epoch: Iteration # 900 error is: 0.05889082
## Epoch: Iteration # 1000 error is: 0.0588387
## Performing Kmeans.
## Performing t-SNE.
## Epoch: Iteration # 100 error is: 10.36092
## Epoch: Iteration # 200 error is: 1.167142
...
## Epoch: Iteration # 900 error is: 0.793667
## Epoch: Iteration # 1000 error is: 0.6030175
```

To assess the performance of our method, we compute the normalized mutual information (NMI) between the clusters inferred by SIMLR and the ground truth clusters. NMI takes values between 0 and 1, with higher values reflecting better performance.

```
nmi = compare(BuettnerFlorian$true_labs[,1], example$y$cluster,
              method="nmi")
```

```
print(nmi)
```

```
## [1] 0.888298
```

To visualize the results, we plot the cell populations in Figure 1.

```
plot(example$ydata,  
      col = c(topo.colors(BuettnerFlorian$n_clust))[BuettnerFlorian$true_labs[,1]],  
      xlab = "SIMLR component 1",  
      ylab = "SIMLR component 2",  
      pch = 20,  
      main="SIMLR 2D visualization for BuettnerFlorian")
```

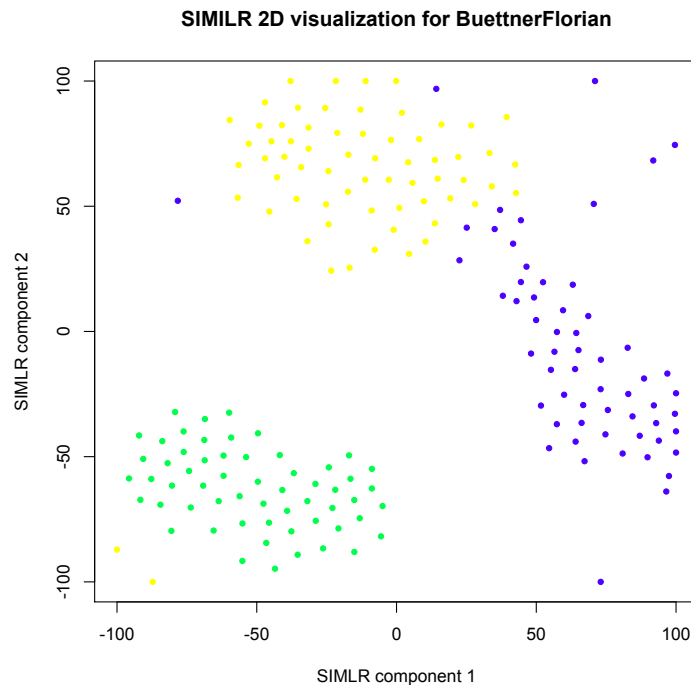


Figure 1: Example of visualization by SIMLR for the BuettnerFlorian dataset.

We also run SIMLR feature ranking on the same inputs to get a rank of the key genes with the related pvalues.

```
set.seed(11111)  
ranks = SIMLR_Feature_Ranking(A=BuettnerFlorian$results$S,  
                              X=BuettnerFlorian$in_X)  
head(ranks$pval)
```

```
## [1] 2.201015e-125 2.531379e-90 5.632172e-77 6.719501e-76
      4.444251e-72 8.822900e-69
head(ranks$aggR)
## [1] 5701 1689 7549 57 2653 8081
```

SIMLR supports SCESet objects [4]. We now create an example object and then run SIMLR on it.

```
library(scran)
ncells = 100
ngenes = 50
mu <- 2^runif(ngenes, 3, 10)
gene.counts <- matrix(rnbinom(ngenes*ncells, mu=mu, size=2),
                      nrow=ngenes)
rownames(gene.counts) = paste0("X", seq_len(ngenes))
sce = newSCESet(countData=data.frame(gene.counts))
output = SIMLR(X = sce, c = 8, cores.ratio = 0)
## X is and SCESet, converting to input matrix.
## Computing the multiple Kernels.
## Performing network diffusion.
## Iteration: 1
## Iteration: 2
...
## Iteration: 9
## Iteration: 10
## Performing t-SNE.
## Epoch: Iteration # 100 error is: 0.3419302
## Epoch: Iteration # 200 error is: 0.0724342
...
## Epoch: Iteration # 900 error is: 0.1660576
## Epoch: Iteration # 1000 error is: 0.02173117
## Performing Kmeans.
## Performing t-SNE.
## Epoch: Iteration # 100 error is: 19.82644
## Epoch: Iteration # 200 error is: 2.559469
...
## Epoch: Iteration # 900 error is: 2.732914
## Epoch: Iteration # 1000 error is: 1.07465
```

We now provide an example application of large-scale SIMLR to an input dataset (a reduced version of the dataset provided in Zeisel, Amit, et al). The full dataset has 9 cell populations, but for the sake of this example, we use a reduced version with only 2 clusters.

```
set.seed(11111)
example_large = SIMLR_Large_Scale(X = ZeiselAmit$in_X,
                                  c = ZeiselAmit$n_clust)
```



```

## Performing fast PCA.
## Performing k-nearest neighbour search.
## Computing the multiple Kernels.
## Performing the iterative procedure 5 times.
## Iteration: 1
## Iteration: 2
## Iteration: 3
## Iteration: 4
## Iteration: 5
## Performing Kmeans.
## Performing t-SNE.
## The main loop will be now performed with a maximum of
## 300 iterations.
## Performing iteration 1.
## Performing iteration 2.
...
## Performing iteration 299.
## Performing iteration 300.

```

Once again, we evaluate the performance of SIMLR by computing the NMI between large-scale SIMLR's inferred clusters and the ground truth clusters.

```

nmi_large = compare(ZeiselAmit$true_labs[,1],example_large$y$cluster,
                    method="nmi")
print(nmi_large)

## [1] 0.9348853

```

We plot the cell populations in Figure 2.

```

plot(example_large$ydata,
     col = c(topo.colors(ZeiselAmit$n_clust))[ZeiselAmit$true_labs[,1]],
     xlab = "SIMLR component 1",
     ylab = "SIMLR component 2",
     pch = 20,
     main="SIMLR 2D visualization for ZeiselAmit")

```

2 Matlab

2.1 Installation

The Matlab version of SIMLR is available on *Github* at <https://github.com/BatzoglouLabSU/SIMLR> (SIMLR branch). The tool may be installed as follows.

```

%%% Installation
% Before running SIMLR, the user needs to mex compile several C-mex files.

```

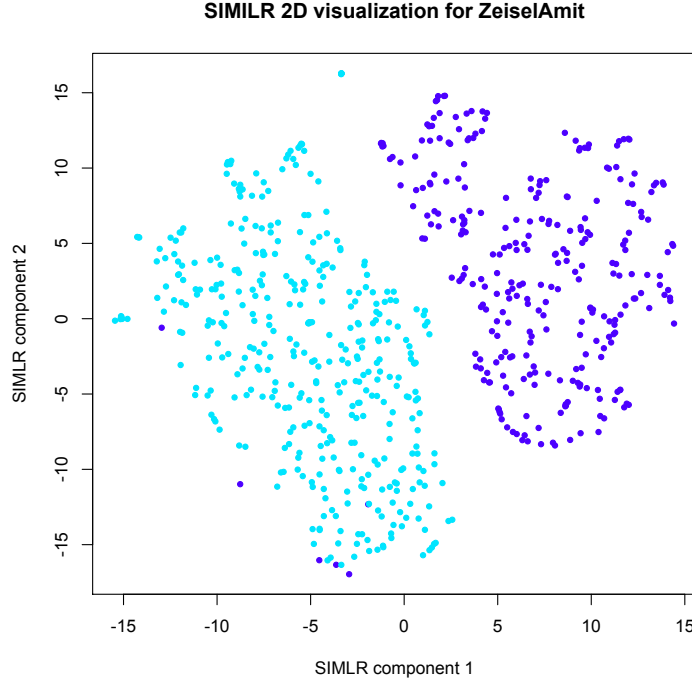


Figure 2: Example of visualization by SIMLR for the ZeiselAmit dataset.

% You can simply run the code INSTALL_SIMLR.m

INSTALL_SIMLR.m

2.2 Examples

We now provide code to run the Matlab implementation of SIMLR on two examples: one for standard SIMLR and one for the large-scale implementation. The code and datasets are available on *Github*.

*%% For small-scale Single-cell RNA-seq with less than 3000 cells,
 %% please use SIMLR.m
 % Given the gene expression matrix in_X, (if in_X is the gene counts
 % matrix, please use log10(1+in_X) first), we can run SIMLR as follows:*

`[y, S, F, ydata] = SIMLR(in_X,C,K);`

*%% Inputs are as follows:
 % in_X is the input gene expression matrix of size N x M, where N is*

```

% the number of cells and M is the number of genes.
% C is the number of clusters,
% K is the number of neighbors, and by default, K =10

%% The outputs are as follows:
% y is the obtained labels
% S is the learned similarity of size  $N \times N$ , where N is the number of cells
% F is the latent variables of size  $N \times C$ 
% ydata is the 2-D visualization of cells

%% You can plot the visualization as follows:
scatter(ydata(:,1),ydata(:,2));

% or if you have any labels, you can color-code the cells and show the
% visualization as follows:
SIMLR_DisplayVisualization(ydata,true_labs);

%% Once you have your similarity S, you can run feature selection as follows:
aggR = SIMLR_Feature_Ranking(S,in_X);

% aggR is a vector of ranking for M genes. Usually we take the top 100 genes as
% the most important/differential genes.

%% For large-scale single-cell RNA-seq data with more than 3000 cells,
%% we recommend using SIMLR_LARGE.m as follows:

% Step 0: If the input is gene counts, we take log10 transformations:
in_X = log10(1+in_X);

% Step 1: Learn the similarity S and the latent embedding F
[S, F] = SIMLR_LARGE(in_X,C,K); % K is usually set to be 30~50 for large scale

% Step 2: Running clustering on F:
y = litekmeans(F,C,'Replicates',50);

% Step 3: Running visualization from S:
% d is the dimension for the visualization, with a default value of 2
ydata = SIMLR_embedding_tsne(S,1,d,F(:,1:2));

% Step 3.1: Show your visualization
SIMLR_DisplayVisualization(ydata,true_labs)

```

References

- [1] Florian Buettner, Kedar N Natarajan, F Paolo Casale, Valentina Proserpio, Antonio Scialdone, Fabian J Theis, Sarah A Teichmann, John C Marioni,

- and Oliver Stegle. Computational analysis of cell-to-cell heterogeneity in single-cell rna-sequencing data reveals hidden subpopulations of cells. *Nature biotechnology*, 33(2):155–160, 2015.
- [2] Amit Zeisel, Ana B Muñoz-Manchado, Simone Codeluppi, Peter Lönnerberg, Gioele La Manno, Anna Juréus, Sueli Marques, Hermany Munguba, Liqun He, Christer Betsholtz, et al. Cell types in the mouse cortex and hippocampus revealed by single-cell rna-seq. *Science*, 347(6226):1138–1142, 2015.
- [3] Gabor Csardi and Tamas Nepusz. The igraph software package for complex network research. *InterJournal*, Complex Systems(1695), 2006.
- [4] Aaron Lun and Karsten Bach. *scraper: Methods for Single-Cell RNA-Seq Data Analysis*, 2016. R package version 1.0.4.